

Technical report of Raspberry Pi prototype for lectures in public universities

Tiago da Silva Almeida¹

¹ Universidade Federal do Tocantins, Computer Science, Tocantins, Brazil

Reception date of the manuscript: 11/02/2020

Acceptance date of the manuscript: 27/02/2020

Publication date: 09/03/2020

Abstract— This paper presents a low-cost prototype for lectures in public universities based on Raspberry Pi. It is described how to connect and configure the prototype using an infrared remote control. Technologies applied in education are widely explored in recent researches and can be held to lower the cost. The proposed prototype is 86% cheaper on average (compared to ordinary computer) and can be used for automation of the classrooms besides lectures. For example, the prototype can be additionally used to access control, environment monitoring, and management of the environment utilization by the users.

Keywords—Raspberry Pi, Prototype, Lectures

I. INTRODUCTION

With the advent of the Internet of Things (IoT), System-on-Chip (SoC) devices are widely used in many applications for different areas. SoCs can be used from military, medical, commerce, logistics, entertainment and educational applications.

In many educational institutions in Brazil, the allocation of resources for the development of activities is a bottleneck. This directly impacts the quality of the applied activities. Thus, low-cost resources are important to the maintenance of the activities.

For the simple activity of lectures, the most used equipment is a multimedia projector and a computer. At public universities, costs on average R\$ (Reais) 5,518.12¹ and computer costs on average R\$ 4,369.00². This is a high cost for public universities in periods of limited spending.

Many research papers have been done using Raspberry Pi due to its facility to work and implement solutions. For example, [1] describes a data processing system based on Raspberry Pi to inertial sensors. The paper is just an introduction and does not bring any results. [2] developed a touchscreen platform for rodent testing to allow cognitive testing based on

Raspberry Pi and Arduino. This has advantages compared to the standard maze apparatuses typically employed in rodent behavioral testing, according to the authors.

[3] proposed the extension of the AP (Access Point) configuration algorithm to deal with this dynamic nature (users often repeat joining and leaving the network) using Raspberry Pi for the AP. [4] proposed a renewable energy monitoring system for data acquisition and transmission applied to real-time cloud monitoring of a decentralized photovoltaic plant also based on Raspberry Pi embedded in each plant.

In the same field, [5] used a Raspberry Pi as a web server for home automation, with all transducers (sensors and actuators) connect by the internet to the Raspberry Pi.

On the other side, a lot of papers focus on “how to use new technologies or devices to improve education and solve its issues”. For example: [6], [7], [8], [9], [10], [11], [12], [13], [14], [15] and [16].

In this context, this paper proposes a prototype for uses in lectures based on Raspberry Pi. The prototype was built based on the work of [17]. Section II is explained how the prototype was connected and configured. Section III concludes the paper and makes some discussion about the prototype.

II. METHODS AND MATERIALS

The materials and devices used for the prototype were:

- 1 -A Raspberry Pi 3 B with an 32 GB SD card (R\$ 360,00);
- 2 - Li-ion Battery 2.200 mAh (R\$ 50,00 each);
- 1 - TP4056 Battery Charger Module (R\$ 6,00);

Contact data: Tiago da Silva Almeida, Quadra 109 Norte, Avenida NS-15, ALCNO-14, Plano Diretor Norte, Palmas/TO, Brazil, tiagoalmeida@uft.edu.br

¹According to data from the Brazilian federal government portal, available for consultation at: <http://paineldeprescos.planejamento.gov.br/analise-materiais>.

²Also according to data from the Brazilian federal government portal, available for consultation at: <http://paineldeprescos.planejamento.gov.br/analise-materiais>.

- 1 - 1838B IR receiver (R\$ 7,00);
- 1 - Universal Remote for datashow (R\$ 130,00).

The IR (InfraRed) sensor has just three pins, which was connect with three pins on the GPIO (General-Purpose Input Output) connector in Raspberry Pi (VCC, GND, and SIGNAL). The connection is: a) VCC pin of IR receiver to 3.3V pin of Raspberry Pi; b) GND pin of IR receiver to any Ground pin of Raspberry Pi; c) SIGNAL pin to GPIO 18 of Raspberry Pi.

Each button of an IR remote control has a string of specific encoding. When a button is pressed, the IR transmitter in the remote control was send out the corresponding IR encoding signals. On the other side, when the IR receiver receives certain encoding signals, it will decode them to identify which button is pressed.

To this prototype, it was used the LIRC³ library to read infrared signals returned by buttons of the remote control and translate them to button values. Then, it was used pylirc (Python) to simplify the process for reading values from the remote control.

It's important to highlight that LIRC library runs without problems with Raspbian Jessie operational system until 2017. For newer versions of Raspbian is required other configuration of the libraries.

To configure the Raspberry Pi for remote control is required these steps:

1. Download the LIRC library using the following command from terminal:

```
1 sudo apt-get install lirc
```

2. Add the two lines below to /etc/modules. This will start the modules on boot of Raspbian. Pin 18 will be used to take the output from the IR sensor. Use the following command:

```
1 sudo nano /etc/modules
```

And add this two lines in the file opened:

```
1 lirc\_dev
2 lirc\_rpi gpio\_in\_pin=18
```

Also, edit the /boot/config.txt file. Open the file using the command:

```
1 sudo nano /boot/config.txt
```

And add the following line to the file:

³LIRC is a package that allows you to decode infrared signals of many (but not all) commonly used remote controls. LIRC runs as a daemon that will decode IR signals received by the device drivers and provide the information on a socket. We will then write a program in the user space to monitor this socket for input events using the LIRC client library.

```
1 dtoverlay=lirc-rpi,gpio\_in\_pin=18
```

3. Edit /etc/lirc/hardware.conf using the command:

```
1 sudo nano /etc/lirc/hardware.conf
```

And type the following code exactly as shown below.

```
1 # /etc/lirc/hardware.conf
2 #
3 # Arguments which will be used when launching
4   lircd
5 LIRC_ARGS="--uinput"
6 # Don't start lircmd even if there seems to be
7   a good config file
8 # START_LIRCMD=false
9 # Don't start irexec, even if a good config
10  file seems to exist.
11 # START_IEXEC=false
12 # Try to load appropriate kernel modules
13 LOAD_MODULES=true
14 # Run "lircd --driver=help" for a list of
15   supported drivers.
16 DRIVER="default"
17 # usually /dev/lirc0 is the correct setting
18   for systems using udev
19 DEVICE="/dev/lirc0"
20 MODULES="lirc_rpi"
21 # Default configuration files for your
22   hardware if any
23 LIRC_CONF=""
24 LIRCMD_CONF=""
```

Type the command to reboot.

4. To perform a quick test to check if LIRC is working, it is required to stop the LIRC daemon and start mode2. mode2 shows the pulse/space length of infrared signals.

To start the LIRC in mode2 type the following command:

```
1 sudo /etc/init.d/lirc stop
2 mode2 -d /dev/lirc0
```

5. The irrecord command helps to discover the IR codes used by the remote control and assist with creating a lircd.conf file which will be used by LIRC.

To record the IR code type the following command from the terminal:

```
1 irrecord -d /dev/lirc0 ~/lircd.conf
```

Once started, irrecord is going to show detailed instructions on how to configure the remote control. Each button should be a pre-defined name. Running irrecord-list-namespace is going to display a list of available names. It can be set any name for any button from the irrecord-list-namespace list. After successfully recording the configuration file, it looks like the following listing:

```

1 # Please make this file available to others
2 # by sending it to <lirc@bartelmus.de>
3 #
4 # this config file was automatically generated
5 # using lirc -0.9.0-pre1(default) on Mon Dec 16
6 #       20:26:03 2019
7 #
8 # contributed by
9 #
10 # brand: /home/pi/lircd.conf
11 # model no. of remote control:
12 # devices being controlled by this remote:
13 #
14 begin remote
15
16 name /home/pi/lircd.conf
17 bits 16
18 flags SPACE_ENC|CONST_LENGTH
19 eps 30
20 aeps 100
21
22 header 9093 4490
23 one 645 1625
24 zero 645 494
25 ptrail 647
26 pre_data_bits 16
27 pre_data 0x8C73
28 gap 108500
29 toggle_bit_mask 0x0
30
31 begin codes
32 KEY_LEFT 0x13EC
33 KEY_UP 0x639C
34 KEY_RIGHT 0x936C
35 KEY_DOWN 0xE31C
36 KEY_ENTER 0x53AC
37 KEY_TAB 0xE11E
38 KEY_ALTRELEASE 0x31CE
39 KEY_ALTPRESS 0xF10E
40 KEY_CLOSE 0x916E
41 KEY_PRESENTATION 0xD32C
42 KEY_ESC 0x33CC
43 KEY_POWER 0x817E
44 end codes
45
46 end remote

```

Now replace the existing configuration file (which is most likely empty) with the created new one, by the following command:

```

1 sudo cp lircd.conf /etc/lirc/lircd.conf

```

6. Restart LIRC by the following command:

```

1 sudo /etc/init.d/lirc restart

```

irw can be used to test the new configuration file. irw sends data from Unix domain socket to stdout.

7. At this stage was created a virtual keystroke from the remote control. It was used the Python (pylirc) code to do this. pylirc is LIRC Python wrapper and it is required to access LIRC from Python programs. To install pylirc should be typed following command in the terminal:

```

1 sudo apt-get install python-pylirc

```

Create a lircrc.conf file. The lircrc file is used to map the key symbols defined in lircd.conf to application-specific strings. Thus, this file cannot be configured until lircd has been configured to provide proper key symbols as displayed by irw. A lircd was already configured in the previous step.

The lircrc file should be placed in the home directory as ~/.config/lircrc. Optionally you can create a system-wide configuration file located in /etc/lirc/lircrc which will be used when no lircrc file can be found in the /home directory.

The syntax of the lircrc file consists of one or more of the following constructions:

```

1 begin
2 prog = ...
3 remote = ...
4 button = ...
5 [button = ...] (optional, for key sequences)
6 repeat = ...
7 delay = ...
8 ignore_first_events = ...
9 config = ...
10 [config = ...] (optional, for toggle button
11     behaviour)
12 mode = ...
13 flags = ...
14 end

```

The complete lircrc file that was prepared for the prototype is as follows:

```

1 begin
2 button = KEY_ESC
3 prog = irexec
4 repeat = 0
5 config = esc
6 end
7 begin
8 button = KEY_TAB
9 prog = irexec
10 repeat = 0
11 config = tab
12 end
13 begin
14 button = KEY_LEFT
15 prog = irexec
16 repeat = 0
17 config = left
18 end
19 begin
20 button = KEY_RIGHT
21 prog = irexec
22 repeat = 0
23 config = right
24 end
25 begin
26 button = KEY_DOWN
27 prog = irexec
28 repeat = 0
29 config = down
30 end
31
32 begin
33 button = KEY_UP

```

```

34 prog = irexec
35 repeat = 0
36 config = up
37 end
38 begin
39 button = KEY_ALTRELEASE
40 prog = irexec
41 repeat = 0
42 config = altrelease
43 end
44 begin
45 button = KEY_ALTPRESS
46 prog = irexec
47 repeat = 0
48 config = altpress
49 end
50 begin
51 button = KEY_CLOSE
52 prog = irexec
53 repeat = 0
54 config = close
55 end
56 begin
57 button = KEY_PRESENTATION
58 prog = irexec
59 repeat = 0
60 config = presentation
61 end
62 begin
63 button = KEY_POWER
64 prog = irexec
65 repeat = 0
66 config = power
67 end

```

8. PyUserInput is a cross-platform Python module to take control of the mouse and keyboard in Python. PyUserInput is registered on PyPI (Python Package Index) and updated periodically, so tools such as pip can be used to install. Type the following command from terminal:

```
1 pip install PyUserInput
```

Here is the complete python program to emulate keyboard from remote key-press:

```

1 #!/usr/bin/env python
2
3 import pylirc, time
4 from pykeyboard import PyKeyboard
5 from pymouse import PyMouse
6 import os
7
8 k = PyKeyboard()
9 m = PyMouse()
10
11 blocking = 0;
12
13 x = 0
14 y = 0
15
16 def setup():
17     pylirc.init("irexec")
18
19 def key_press(config):
20     if config == 'esc':
21         k.tap_key(k.escape_key)
22         print 'ESC key'
23     if config == 'tab':

```

```

24         k.tap_key(k.tab_key)
25         print 'TAB key'
26     if config == 'presentation':
27         k.press_key(k.control_key)
28         k.press_key(k.shift_key)
29         k.tap_key('p')
30         k.release_key(k.shift_key)
31         k.release_key(k.control_key)
32         print 'Presentation mode in Okular'
33     if config == 'power':
34         os.system('sudo shutdown now')
35         print 'Power Down'
36     if config == 'close':
37         k.press_key(k.control_key)
38         k.tap_key('q')
39         k.release_key(k.control_key)
40         print 'Closing Okular'
41     if config == 'altrelease':
42         k.release_key(k.alt_key)
43         print 'Releasing ALT'
44     if config == 'altpress':
45         k.press_key(k.alt_key)
46         print 'Pressing ALT'
47
48 def loop():
49     while True:
50         s = pylirc.nextcode(1)
51         while(s):
52             for (code) in s:
53                 print 'Command: ', code["config"]
54                 key_press(code["config"])
55                 if(not blocking):
56                     s = pylirc.nextcode(1)
57             else:
58                 s = []
59
60 def destroy():
61     pylirc.exit()
62
63 if __name__ == '__main__':
64     try:
65         setup()
66         loop()
67     except KeyboardInterrupt:
68         destroy()

```

9. Save the file as remote.py to the directory /home/pi in the Raspberry Pi. Make the file executable by the following command:

```
1 sudo chmod +x remote.py
```

10. Open the autostart file by the command:

```
1 sudo nano /.config/lxsession/LXDE-pi/autostart
```

Add the following two lines at the end of the file:

```

1 @pcmanfm /home/pi/media —wallpaper—mode=tile
2 @python /home/pi/remote.py

```

So far, the flash drive with the PDF presentation is going to load when the Raspbian startup. To do so, it is automatic open up the /home/pi/media directory to choice the flash drive. The presentation file is open with Okular PDF reader and it is pre-installed in Raspberry Pi. At the same time, it is also started remote.py file.

TABLE 1: POWER SPECIFICATION OF ALL RASPBERRY PI MODELS.

Product	Recommended PSU current capacity	Maximum total USB peripheral current draw	Typical bare-board active current consumption
Raspberry Pi Model A	700mA	500mA	200mA
Raspberry Pi Model B	1.2A	500mA	500mA
Raspberry Pi Model A+	700mA	500mA	180mA
Raspberry Pi Model B+	1.8A	600mA/1.2A (switchable)	330mA
Raspberry Pi 2 Model B	1.8A	600mA/1.2A (switchable)	350mA
Raspberry Pi 3 Model B	2.5A	1.2A	400mA
Raspberry Pi 3 Model A+	2.5A	Limited by PSU, board, and connector ratings only.	350mA
Raspberry Pi 3 Model B+	2.5A	1.2A	500mA
Raspberry Pi 4 Model B	3.0A 1.2A	600mA	
Raspberry Pi Zero W/WH	1.2A	Limited by PSU, board, and connector ratings only.	150mA
Raspberry Pi Zero	1.2A	Limited by PSU, board, and connector ratings only	100mA

To add the battery inside of a case of the Raspberry Pi, the default micro USB (Universal Serial Bus) port is not a good option. A good option is providing power using GPIO. For this, it was used the GPIO of Raspberry Pi of VCC (pin 4) and GND (pin 6). The power specification of all Raspberry Pi models are shown in Table 1⁴.

For tests were connected to two batteries of 3.7 V in series, adding voltage to 7.4 V and keep the current. How the batteries have 2,200 mAh, the prototype was able to supply the system for 2.5 hours. It is a low time supply for four hour-lecture (50 minutes). But, adding two more batteries of the same features is enough for almost 5 hours of running.

III. CONCLUSION

This paper is a technical report about the construction of a low-cost prototype to use in lectures by universities based on Raspberry Pi. Due to the simplicity, the prototype runs well and it is easy to implement.

The Raspberry Pi was chosen because of its capability of processing, besides the low cost. The prototype can be installed in classrooms and uses to control the environment, like temperature monitoring and access control of the classrooms. It can be used as a web server, or such as a node in a local network.

At the end, the total costs were R\$ 603.00 on average, against R\$ 4,369.00 of an ordinary computer in public universities, also on average. This represents 86.19% of cost reduction just for regular lectures, besides other possibilities of implementation using the same equipment.

REFERENCES

- [1] A. Alvarelos, A. Vázquez, and J. Rabuñal, "Raspberry pimiu: Raspberry pi based inertial sensor data processing system," *Proceedings*, vol. 2, no. 18, 2018. [Online]. Available: <https://www.mdpi.com/2504-3900/2/18/1159>
- [2] J. D. O'Leary, O. F. O'Leary, J. F. Cryan, and Y. M. Nolan, "A low-cost touchscreen operant chamber using a raspberry pi™," *Behavior Research Methods*, vol. 50, no. 6, pp. 2523–2530, Dec 2018. [Online]. Available: <https://doi.org/10.3758/s13428-018-1030-y>
- [3] M. M. Islam, N. Funabiki, M. Kuribayashi, S. K. Debnath, K. I. Munene, K. S. Lwin, R. W. Sudibyo, and M. S. A. Mamun, "Dynamic access-point configuration approach for elastic wireless local-area network system and its implementation using raspberry pi," *International Journal of Networking and Computing*, vol. 8, no. 2, pp. 254–281, 2018.
- [4] R. I. Pereira, I. M. Dupont, P. C. Carvalho, and S. C. Jucá, "Iot embedded linux system based on raspberry pi applied to real-time cloud monitoring of a decentralized photovoltaic plant," *Measurement*, vol. 114, pp. 286–297, 2018. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S026322411730605X>
- [5] V. Vujović and M. Maksimović, "Raspberry pi as a sensor web node for home automation," *Computers & Electrical Engineering*, vol. 44, pp. 153–171, 2015. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S0045790615000257>
- [6] N. Miglani and P. Burch, "Educational technology in india: The field and teacher's sensemaking," *Contemporary Education Dialogue*, vol. 16, no. 1, pp. 26–53, 2019. [Online]. Available: <https://doi.org/10.1177/0973184918803184>
- [7] G. Friedland, W. Hürst, and L. Knipping, "The future of multimedia education and educational multimedia," in *Proceedings of the International Workshop on Educational Multimedia and Multimedia Education*, ser. Emme '07. New York, NY, USA: Association for Computing Machinery, 2007, p. 125–126. [Online]. Available: <https://doi.org/10.1145/1290144.1290167>
- [8] S. Xu, T. Hu, and Q. Zhou, "A case study in jiuwuji minority middle school on rural teachers' educational technology training in terms of a switch of educational product," in *2011 IEEE 3rd International Conference on Communication Software and Networks*, May 2011, pp. 318–320.
- [9] M. Qian, B. Zhao, and Y. Gao, "Exploring the training path of design thinking of students in educational technology," in *2019 IEEE International Conference on Computer Science and Educational Informatics (CSEI)*, Aug 2019, pp. 315–319.
- [10] J. Hongyan and R. Jianfeng, "A brief analysis on the seminar of professional construction for modern preschool educational technology," in *2012 2nd International Conference on Consumer Electronics, Communications and Networks (CECNet)*, April 2012, pp. 2262–2264.
- [11] A. I. Taganov, A. N. Kolesenkov, D. A. Perepelkin, and D. S. Zhuravlev, "Management of educational risk on the basis of data mining in gis," in *2017 International Conference "Quality Management, Transport and Information Security, Information Technologies" (IT QM IS)*, Sep. 2017, pp. 577–580.
- [12] C. Wei and L. Yuan, "Reflection on college informationized teaching model under the background of educational informatization," in *2019 IEEE International Conference on Computer Science and Educational Informatics (CSEI)*, Aug 2019, pp. 81–83.
- [13] G. R. Iksanova, "Improvement of university education and educational service consumer interconnection in russia," in *2013 International Conference on Interactive Collaborative Learning (ICL)*, Sep. 2013, pp. 666–668.

⁴<https://www.raspberrypi.org/documentation/faqs/>.

- [14] S. Tatiana Alexandrovna, "The role of distance educational technologies in the development of educational programmes in a network form: Experience and relevant problems," in *2018 IV International Conference on Information Technologies in Engineering Education (Inforino)*, Oct 2018, pp. 1–4.
- [15] H. M. Fardoun, A. S. Mashat, and L. Gonzalez, "New subject to improve the educational system: Through a communication channel between educational institution-company," in *2013 Federated Conference on Computer Science and Information Systems*, Sep. 2013, pp. 709–712.
- [16] T. A. Tabishev, M. V. Alikaeva, and A. L. Betuganova, "Electronic informational and educational environment and organization of the educational process of a modern university (on the materials of the kabardino-balkar state university)," in *2019 International Conference "Quality Management, Transport and Information Security, Information Technologies" (IT QM IS)*, Sep. 2019, pp. 569–572.
- [17] Taifur. (2011) Raspberry pi presentation machine with remote control. Tomado de <https://www.instructables.com/id/Raspberry-Pi-Presentation-Machine-With-Remote-Cont/>.